



# **PERTEMUAN XI**

## **OPERASI FILE**



# JENIS FILE

- **File Biner** : file yang pola penyimpanan di dalam disk berbentuk biner, yaitu seperti bentuk pada memori RAM (komputer).  
Dipakai untuk menyimpan data kompleks, mis : struct.
- **File Teks** : file yang pola penyimpanan datanya dalam bentuk karakter.  
Dipakai untuk menyimpan data seperti karakter atau string.
- **Penentuan mode teks dan mode biner :**
  - t untuk mode teks
  - b untuk mode biner

**Contoh :**

"rt" : mode file adalah teks dan file hendak dibaca

"rt+" : mode file adalah teks dan file bisa dibaca dan ditulisi.

Bisa juga ditulis : "r+t"

"rb" : mode file adalah biner dan file hendak dibaca.

# OPERASI BACA & TULIS

## FILE PER INT

---

- Perintah yang digunakan : `_putw ()`, `_getw()`.
- Bentuk deklarasi :

```
int _putw(int nilai, FILE *ptr_file);  
int _getw(FILE *ptr_file);
```

Kegunaan :

`_getw()` untuk membaca sebuah data bertipe `int` dari file  
`_putw()` untuk menyimpan sebuah data bertipe `int` ke file.

# CONTOH PROGRAM TULIS

```
#include <stdio.h>
#include <stdlib.h>
main( )
{
    FILE *pf;           /* ptr-ke-FILE */
    int nilai, data, i;
    char jawab;
    if((pf=fopen("BILANGAN.DAT", "wb")) == NULL )
    {
        printf("file gagal diciptakan!\n");
        exit(1);
    }
    printf ("Masukkan banyaknya data : ");
    scanf ("%d",&data);
    for (i=0;i<data;i++) {
        printf("\nBilangan yang disimpan: ");
        scanf("%d", &nilai);           /* baca nilai dr keyboard */
        _putw(nilai, pf);              /* baca bilangan ke file */
    }
    printf("\nOke. Data sudah disimpan dalam file.\n");
    fclose(pf);                       /* menutup file */
}
```

Masukkan banyaknya data : 3

Bilangan yang disimpan : 70

Bilangan yang disimpan : 80

Bilangan yang disimpan : 90

# CONTOH PROGRAM BACA

```
#include <stdio.h>
#include <stdlib.h>
main()
{
    FILE *pf;          /* ptr ke file */
    int nilai, nomor = 0;
    /* Buka file biner untuk dibaca */
    if((pf=fopen("BILANGAN.DAT","rb")) == NULL)
    {
        printf("File gagal dibuka.\n");
        exit(1);
    }
    printf("Isi file BILANGAN.DAT : \n");
    while(1)          /* file berhasil dibuka */
    {
        nilai = _getw(pf); /* Baca sebuah int dr file */
        if (feof(pf) != 0) break; /* Jika akhir file, keluar loop*/
        printf("%2d. %d \n", ++nomor, nilai); /* Tampilkan ke layar */
    }

    fclose(pf);      /* Tutup file */
}
```

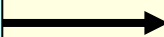
Isi file BILANGAN.DAT :

1. 70

2. 80

3. 90

feof : untuk mendeteksi akhir file



# OPERASI BACA & TULIS

## FILE PER BLOK

---

- Fungsi : untuk menyimpan atau membaca data file dalam bentuk kesatuan blok (sejumlah byte), misal float atau struct.
- Perintah yang digunakan : `fread ()` dan `fwrite ()`;
- Bentuk deklarasi :

```
int fread(void *buffer, int n, FILE *ptr_file);  
int fwrite(void *buffer, int jum_byte, int n, FILE *ptr_file);
```

dengan :

**buffer** : pointer yang menunjuk ke alamat memori

**jum\_byte** : jumlah byte yang akan dibaca atau disimpan

**n** : byknya blok data berukuran **jum\_byte** yg akan ditulis / dibaca

**ptr\_file** : pointer-ke-FILE yang berisi nilai keluaran dari *fopen()*.

# CONTOH PROGRAM TULIS

```
#include <stdio.h>
#include <stdlib.h>

main()
{
    FILE *f_struktur;
    char jawaban;
    int sudah_benar;

    struct {
        char judul[20];
        char pengarang[20];
        int jumlah;
    } buku;          /* variabel buku bertipe struktur */

    if((f_struktur = fopen("DAFBUKU.DAT", "wb")) == NULL)
    {
        printf("File tidak dapat diciptakan !\n");
        exit(1);
    }
}
```

# CONTOH PROGRAM TULIS – CONT. 1

---

```
do {  
    fflush(stdin); /* Hapus isi penampung keyboard */  
    printf("Judul buku                : ");  
    gets(buku.judul);  
    printf("Nama pengarang           : ");  
    gets(buku.pengarang);  
    printf("Jumlah buku                : ");  
    scanf("%d", &buku.jumlah);  
    fflush(stdin); /* Hapus isi penampung keyboard */  
  
    /* Rekam sebuah data bertipe struktur */  
    fwrite(&buku, sizeof(buku), 1, f_struktur);  
    printf("\nMau merekam data lagi [Y/T] ?");  
    jawaban = getchar();  
    printf("\n");  
} while(jawaban == 'Y' || jawaban == 'y');  
fclose(f_struktur);      /* Tutup file */  
}
```

# CONTOH PROGRAM BACA

```
#include <stdio.h>
#include <stdlib.h>

main()
{
    FILE *f_struktur;
    int i=1;
    struct {
        char judul[20];
        char pengarang[20];
        int jumlah;
    } buku;
    /* variabel buku bertipe struktur */

    if((f_struktur = fopen("DAFBUKU.DAT", "rb")) == NULL)
    {
        printf("File tidak dapat dibuka !\n");
        exit(1);
    }
    printf("%2s. %-30s %-30s %s\n\n", "No", "Judul Buku", "Nama Pengarang", "Jumlah");
    /* diulang selama masih ada record yg terbaca dlm file */
    while(fread(&buku, sizeof(buku), 1, f_struktur) == 1)
    printf("%2d. %-30s %-30s %4d\n", i++, buku.judul, buku.pengarang, buku.jumlah);
    printf("\n");
    fclose(f_struktur);
    /* Tutup file */
}
```

Ukuran char-nya harus sama dengan yang di program tulis

/\* variabel buku bertipe struktur \*/

if((f\_struktur = fopen("DAFBUKU.DAT", "rb")) == NULL)

{

printf("File tidak dapat dibuka !\n");  
exit(1);

}

printf("%2s. %-30s %-30s %s\n\n", "No", "Judul Buku", "Nama Pengarang", "Jumlah");

/\* diulang selama masih ada record yg terbaca dlm file \*/

while(fread(&buku, sizeof(buku), 1, f\_struktur) == 1)

printf("%2d. %-30s %-30s %4d\n", i++, buku.judul, buku.pengarang, buku.jumlah);

printf("\n");

fclose(f\_struktur);  
/\* Tutup file \*/

}

Menyatakan data sebanyak 1 x ukuran variabel struct buku

# OPERASI BACA & SIMPAN DATA STRING PADA FILE

- Perintah yang digunakan : *fgets()* dan *fputs()*.
- Bentuk deklarasi :

```
int fputs(char *str, FILE *ptr_file);  
char fgets(char *str, int n, FILE *ptr_file);
```

Kegunaan :

*fputs()* : menyimpan string *str* ke dalam file.

*fgets()* : membaca string dari file sampai ditemukannya karakter baris baru '\n' atau setelah (n-1) karakter, dengan n adalah panjang maksimal string yang dibaca per waktu-baca.

**Note :**

- ❑ ***Saat simpan, fputs()* tidak menambahkan karakter baris-baru ('\n') dengan sendirinya, dan karakter null tidak ikut disimpan.**
- ❑ **Baik *fgets()* maupun *fputs()* digunakan untuk file teks.**

# CONTOH PROGRAM TULIS - 1

```
#include <stdio.h>
#include <stdlib.h>
main( )
{
    FILE *pf;           /* ptr-ke-FILE */
    int data, i;
    char nama[40];
    if((pf=fopen("latihan.txt", "w")) == NULL )
    {
        printf("file gagal diciptakan!\n");
        exit(1);
    }
    printf ("Masukkan banyaknya data : ");
    scanf ("%d",&data);
    for (i=1;i<=data;i++) {
        printf("\nNama ke %d : ",i); fflush(stdin);
        gets(nama);
        strcat(nama,"\n");
        fputs(nama, pf);
    }
    printf("\nOke. Data sudah disimpan dalam file.\n");
    fclose(pf);        /* menutup file */
}
```

# CONTOH PROGRAM TULIS - 2

```
#include <stdio.h>
#include <stdlib.h>
main( )
{
    FILE *pf;                /* ptr-ke-FILE */
    int data, i;
    char nama[40];
    if((pf=fopen("latihan.txt", "w")) == NULL )
    {
        printf("file gagal diciptakan!\n");
        exit(1);
    }
    printf("\nNama ke %d : ",i); fflush(stdin);
    gets(nama);
    fputs(nama, pf);

    printf("\nOke. Data sudah disimpan dalam file.\n");
    fclose(pf);             /* menutup file */
}
```

# CONTOH PROGRAM BACA

```
#include <stdio.h>
#include <stdlib.h>
main( )
{
    FILE *pf;          /* ptr-ke-FILE */
    int data, i;
    char nama[40];
    if((pf=fopen("latihan.txt", "r")) == NULL )
    {
        printf("file gagal dibuka!\n");
        exit(1);
    }
    /*Baca file per string sampai ditemui EOF*/
    while (fgets(nama,6,pf))
        printf ("%s\n",nama);

    fclose(pf);       /* menutup file */
}
```

Hanya  
mencetak 6  
karakter  
per baris

# AKSES FILE BINER SCR ACAK

- Tujuan : membaca data di tengah file scr cepat.
- Perintah yang digunakan : `fseek ()`.
- Bentuk deklarasi :

```
int fseek(FILE *ptr_file, long int offset, int posisi);
```

dengan :

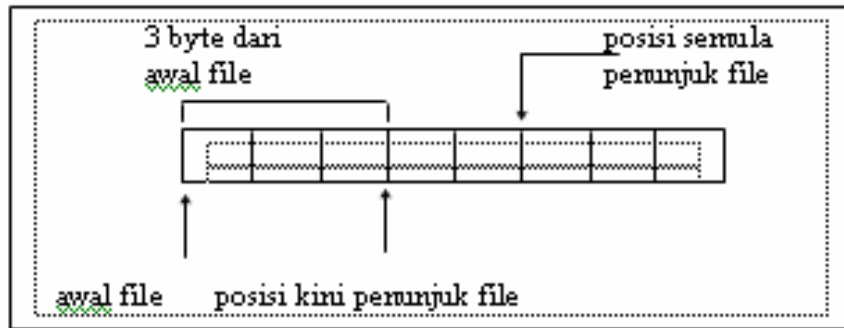
**ptr\_file** adalah pointer yang berasal dari keluaran `fopen()`  
**offset** menyatakan jumlah byte terhadap posisi  
**posisi** dapat diisi suatu nilai yang tertera pada tabel

Konstanta	Nilai	Lokasi file
<code>SEEK_SET</code>	0	Awal file
<code>SEEK_CUR</code>	1	Posisi penunjuk file saat ini
<code>SEEK_END</code>	2	Akhir file

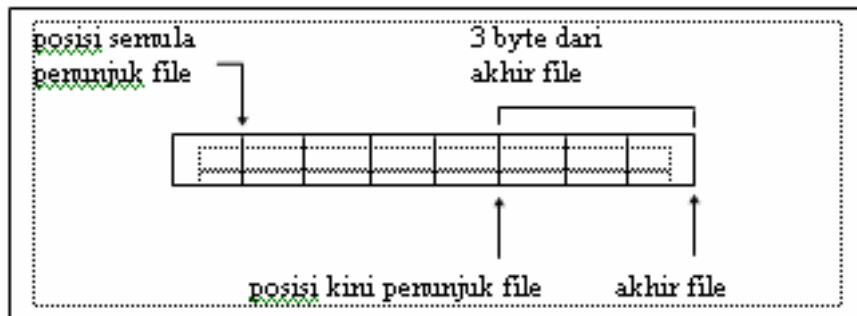
**Prototype : `stdio.h`**

# CONTOH APLIKASI fseek ()

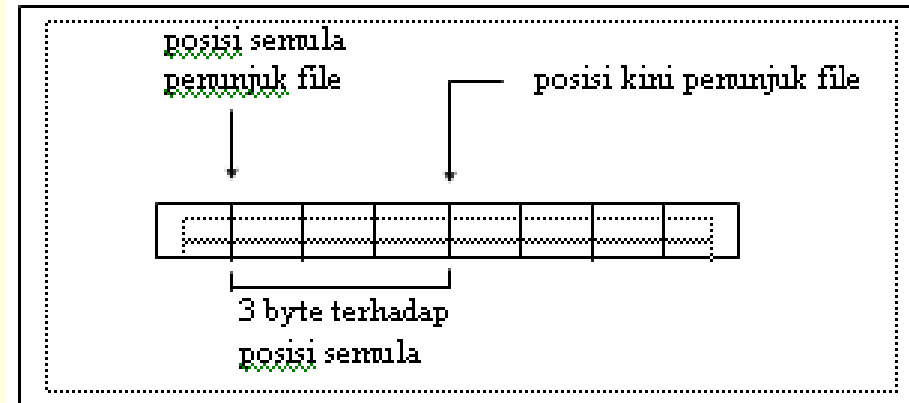
- `fseek(pf, 3, SEEK_SET);`



- `fseek(pf, 3, SEEK_END);`



- `fseek(pf, 3, SEEK_CURR);`



# CONTOH PROGRAM fseek UNTUK CARI KARAKTER

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main()
{
    FILE *pf;
    char kar,jawab;
    int i, no_record;
    long int offset_byte;
    if((pf = fopen("latihan.txt", "r")) == NULL) {
        printf("File tidak dapat dibuka !\n");
        exit(1);
    }
    do
    {
        printf("\n Nomor record dr data yg mau ditampilkan : ");
        scanf("%d", &no_record);
        offset_byte = (no_record-1);
        fseek(pf, offset_byte, SEEK_SET);
        kar=fgetc(pf);
        putchar(kar);
        printf("\nMau mencoba lagi (Y/T)? ");
        jawab=getche();
    } while (jawab == 'y' || jawab == 'Y');
    printf("\n");
    fclose(pf);
}
```

*/\* baca kar dari file \*/  
/\* tampilkan ke layar\*/*

*/\* Tutup file \*/*

# CONTOH PROGRAM fseek UNTUK CARI BILANGAN

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main()
{
    FILE *pf;
    char jawab;
    int i, no_record, nilai;
    long int offset_byte;
    if((pf = fopen("bilangan.dat", "rb")) == NULL) {
        printf("File tidak dapat dibuka !\n");
        exit(1);
    }
    do
    {
        printf ("\n");
        printf("Nomor record dr data yg mau ditampilkan : ");
        scanf("%d", &no_record);
        offset_byte = (no_record-1)*sizeof (int);
        fseek(pf, offset_byte, SEEK_SET);
        nilai = _getw(pf);          /* Baca sebuah int dr file */
        printf("%d \n", nilai);
        printf("\nMau mencoba lagi (Y/T)? ");
        jawab=getche();
    } while (jawab == 'y' || jawab == 'Y');
    printf("\n");
    fclose(pf);                  /* Tutup file */
}
```

# CONTOH PROGRAM fseek UNTUK CARI STRING

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main()
{
    FILE *pf;
    char jawab,nama[20];
    int i, no_record;
    long int offset_byte;
    if((pf = fopen("latihan.txt", "r")) == NULL) {
        printf("File tidak dapat dibuka !\n");
        exit(1);
    }
    do
    {
        printf("\n Nomor record dr data yg mau ditampilkan : ");
        scanf("%d", &no_record);
        offset_byte = (no_record-1);
        fseek(pf, offset_byte, SEEK_SET);
        printf ("%s\n",fgets(nama,20,pf));
        printf("\nMau mencoba lagi (Y/T)? ");
        jawab=getche();
    } while (jawab == 'y' || jawab == 'Y');
    printf("\n");
    fclose(pf);
}
/* Tutup file */
```

# MENGHAPUS FILE

---

- Bentuk deklarasi :

```
int remove (char *namafile);
```

namafile : pointer yang menunjuk ke nama file yang akan dihapus

- ❑ Jika operasi hapus berhasil, akan menghasilkan output = 0.
- ❑ Prototype : stdio.h

# CONTOH PROGRAM HAPUS

---

```
#include <stdio.h>
#include <stdlib.h>
#define PJJG 65
main()
{
    int kode;
    char namafile[PJJG];

    printf("Nama file yang akan dihapus : ");
    gets(namafile);

    kode = remove(namafile);
    if(kode == 0)
        printf("File sudah dihapus\n");
    else
        printf("Gagal dalam menghapus file\n");
}
```

# MENGGANTI NAMA FILE

---

- Bentuk deklarasi :

```
int rename(char *namafilelama, char *namafilebaru);
```

- ❑ Jika operasi hapus berhasil, akan menghasilkan output = 0.
- ❑ Prototype : stdio.h

# CONTOH PROGRAM GANTI NAMA

```
#include <stdio.h>
#include <stdlib.h>
#define PJG 65
main()
{
    int kode;
    char namafilelama[PJG], namafilebaru[PJG];

    printf("Nama file yang akan diganti : ");
    gets(namafilelama);
    printf("Nama file yang baru      : ");
    gets(namafilebaru);

    kode = rename(namafilelama, namafilebaru);
    if(kode == 0)
        printf("Nama file sudah diganti\n");
    else
        printf("Gagal dalam mengganti nama\n");
}
```